Delphi

ประพนธ์ อัศวภาณุวัฒน์

Delphi Technical : ควบคุมจอยสติ๊ก

ใช้โปรแกรม Delphi ควบคุมอุปกรณ์ตัวชี้ คือจอยสติ๊ก เพื่อว่าผู้อ่านและผู้สนใจ นำไปพัฒนาเสริมเข้ากับ โปรแกรมที่มีอยู่

ในปัจจุบันต้องยอมรับว่าการพัฒนาโปรแกรมบนระบบปฏิบัติการวินโดวส์มีความจำเป็นอย่างมาก เนื่องจากสามารถออกแบบระบบ User Interface ให้ง่ายต่อผู้ใช้ได้ และย่อเวลาในการพัฒนาได้เป็นอย่างมาก

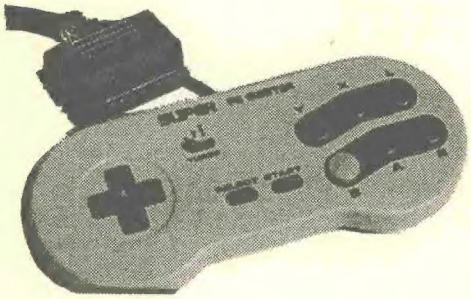
Delphi เป็นภาษาหนึ่งที่มีความนิยมสูงในขณะนี้ รวมทั้งเป็นภาษาที่มีโครงสร้างเหมาะสำหรับการเริ่มต้นที่ดีสำหรับการโปรแกรมแบบ Object Oriented Programming : OOP

ในตอนแรกนี้จะขอกล่าวถึงการใช้โปรแกรม Delphi ควบคุมอุปกรณ์ตัวชี้ (Pointing Device) อีกตัวหนึ่งคือ จอยสติ๊ก (Joystick) เพื่อว่าผู้อ่านและผู้สนใจจะสามารถนำไปพัฒนาเสริมเข้ากับโปรแกรมที่มีอยู่ คงดูเท่ห์ไม่หยอก

ระบบปฏิบัติการวินโดวส์ 95 ได้ออกแบบระบบ Plug And Play มาดีอย่างเหลือเชื่อ (ข้อดีของไมโครซอฟท์เขาล่ะ แตกต่างกับระบบวินโดวส์ 3.x เดิม ซึ่งน่าจะเรียกว่า Plug And Play เสียมากกว่า) ทำให้เราสามารถติดต่อกับอุปกรณ์ดีไวซ์เสริม (Aux. Device) ได้ง่าย ไม่ต้องวิ่งหาไดรเวอร์ (Device Driver) มาติดตั้งให้วุ่นวายใจ

เป็นที่แน่นอนว่า เราจำเป็นต้องมีเจ้าอุปกรณ์ที่เรียกว่าจอยสติ๊กนี้เสียก่อน ซึ่งจะต้องระบุว่า "PC Compatible" หรือใช้กับ PC เท่านั้น เราไม่สามารถนำจอยสติ๊กของเครื่องเล่นเกมอื่น ๆ มาใช้กับคอมพิวเตอร์ของเรา โดยทั่วไปจอยสติ๊กที่ขายอยู่ทั่วไปตามห้องตลาดจะแบ่งออกเป็น 2 ชนิดคือ ชนิดแรกเรียกทับศัพท์ว่า "JoyStick" มีลักษณะเป็นอุปกรณ์สี่เหลี่ยมคล้ายเม้าส์ (Mouse) แต่ตัวใหญ่กว่า สามารถถือหรือวางกับโต๊ะได้ และมีคันโยกเป็นแท่งขึ้นมาจากตรงกลางดังรูปที่ 1 และชนิดที่สองเรียกว่า "JoyPad " ลักษณะเป็นสี่เหลี่ยมผืนผ้า เวลาใช้จะต้องถืออย่างเดียว มีปุ่มลูกศรแทนคันโยกดังรูปที่ 2 ลักษณะคล้ายกับของเครื่องเล่นเกมที่นิยมอยู่ขณะนี้ทั่วไป เช่นเครื่องเล่นเกม PLAY STATION เป็นต้น เราสามารถหาซื้อได้ที่ราคาประมาณ 200-2,500 บาท แล้วแต่รุ่นและประสิทธิภาพ แต่รู้สึกว่าของไมโครซอฟท์จะราคาแพงที่สุด การเลือกซื้อขอแนะนำให้เลือกซื้อรุ่นที่บอกว่าสามารถใช้กับ Windows 95 ได้จะดีกว่า เพราะอาจจะมี Driver และ Tools พิเศษที่สนับสนุนอุปกรณ์ตัวนี้ให้มาด้วย

รูปที่ 1



รูปที่ 2

ขั้นตอนการติดตั้ง

เมื่อเราหาซื้อจอยสติ๊กได้แล้ว ก็ทำการติดตั้งตามคู่มือที่ให้มาหรือตามคำแนะนำของผู้ขาย และติดตั้งซอฟต์แวร์ไดรเวอร์โดยการเรียกโปรแกรม Setup (ในกรณีที่มีโปรแกรมแถมมาให้) หรือเปิดหน้าต่าง Control Panel แล้วเลือกไอคอนรูปจอยสติ๊ก แล้ว

1. เลือกตำแหน่งจอยสติ๊ก ถ้าเป็นตัวแรกหรือมีตัวเดียวคือ "Joystick 1"

2. เลือกชนิดจอยสติ๊กที่ติดตั้ง พยายามเลือกให้ถูกหรือใกล้เคียงที่สุด

3. กดปุ่ม Test แล้วโยกหรือบังคับจอยสติ๊กดูว่าทำงานได้หรือไม่ ถ้าไม่ได้อาจต้องกลับไปตรวจสอบการติดตั้งจอยสติ๊กว่าติดตั้งถูกต้องหรือไม่

4. กดปุ่ม Calibrate แล้วทำตามคำแนะนำ เพื่อตั้งศูนย์ให้จอยสติ๊ก

หลักการ

ใน Delphi เราสามารถควบคุมอุปกรณ์เสริมต่าง ๆ (AUX. Device) ที่ใช้ในส่วนของระบบมัลติมีเดียผ่านยูนิตชื่อ MMSYSTEM. PAS ซึ่งอยู่ในไดเรคทอรี "<Delphi Dir>\Source\Rtl\Win" สามารถระบุการเรียกใช้ไว้ในส่วน Uses ในส่วน Header ของฟอร์ม (Form) ที่เราใช้งานอยู่

การอ่านค่าของจอยสติ๊กจะอ่านจากการเคลื่อนที่ของตำแหน่งแกน X และแกน Y (ในจอยสติ๊กบางรุ่นจะมีแกน Z ด้วย) และจากปุ่มควบคุมทั้ง 4 ปุ่ม (ในรุ่น Extended จะมีมากถึง 32 ปุ่ม ซึ่งจะไม่กล่าวถึงในบทความนี้) เมื่อโยกจอยสติ๊กให้เคลื่อนไหวจะเกิดการเปลี่ยนแปลงค่าบนแกนทั้งสองจะมีค่าอยู่ระหว่าง 0 ถึง 65535 และปุ่มควบคุมก็จะมีค่าเป็น 1, 2, 4, 8 เราสามารถควบคุมได้โดยการกำหนดให้ระบบวินโดวส์ส่งเมสเสจให้กับแอพพลิเคชันของเรา หรือให้ฟังก์ชันของวินโดวส์อ่านค่าพารามิเตอร์ที่เปลี่ยนไปจากเดิมด้วยการตั้ง Timer ควบคุม

การควบคุมจอยสติ๊ก

การควบคุมจอยสติ๊กจำเป็นต้องกำหนดจุดศูนย์กลางและช่วงความเคลื่อนไหว (Movement Range) ของจอยสติ๊ก เราไม่สามารถใช้ค่าที่อ่านได้โดยตรงจากจอยสติ๊ก เพราะค่าอ่านได้มีความแปรปรวนอยู่ตลอดเวลา จึงจำเป็นต้องมีส่วนควบคุมการเคลื่อนไหว และช่วงความละเอียด ดังนี้

● กำหนดจุดศูนย์กลาง และช่วงความเคลื่อนไหว

```
centerX:=32768; centerY:=32768; (* จุดศูนย์กลางของแกน X และ Y *)

notifyRange:=3; (* ช่วงความเคลื่อนไหวมีค่าความละเอียดเท่ากับ 3 Level *)
```

● กำหนดช่วงความเคลื่อนไหว

```
dNegX:=centerX div notifyRange; (* Negative X ค่ามุมซ้ายสุดของแกน X *)

dPosX:=centerX div notifyRange; (* Positive X ค่ามุมขวาสุดของแกน X *)

dNegY:=centerX div notifyRange; (* Negative Y ค่ามุมบนสุดของแกน Y *)

dPosY:=centerX div notifyRange; (* Positive X ค่ามุมล่างสุดของแกน Y *)
```

● การตั้งค่าช่วงความเคลื่อนไหว (Calibrated Joystick)

เนื่องจากจอยสติ๊กของแต่ละบริษัทผู้ผลิตมีรูปร่างต่างกัน ทำให้มีช่วงความเคลื่อนไหวไม่เท่ากัน (ความแปรปรวนต่างกัน) จึงจำเป็นต้องมีการตั้งค่าความเคลื่อนไหวจริงจากการอ่านค่าโดยตรง วิธีที่นิยมทำคือ

1. หาค่าจุดศูนย์กลางของจอยสติ๊ก โดยปล่อยให้ตัวควบคุมทิศทางอยู่ตรงกลาง

2. หาค่ามุมบนซ้ายสุดเพื่อเป็นค่า dNegX และ dNegY โดยการดันตัวควบคุมทิศทางไปด้านบนซ้าย

3. หาค่ามุมล่างขวาสุดเพื่อเป็นค่า dPosX และ dPosY โดยการดันตัวควบคุมทิศทางไปด้านขวาล่าง

โปรแกรม

โปรแกรมในส่วนแรกจะเป็นการแสดงการควบคุมจอยสติ๊กผ่านวินโดวส์เมสเสจ ดังลิสติงที่ 1 ตัวโปรแกรมเขียนด้วย Delphi และเพื่อให้เข้าใจ โดยง่ายขอให้สร้างคอมโปเนนต์และกำหนดค่าพรอพเพอร์ตี้ส์ตามตารางที่ 1

ตารางที่ 1 การกำหนดค่าคอมโปเนนต์ของ Project : JoyMesg

Project Name : JoyMesg.dpr
Unit Name : UjoyMesg.pas
Form Name : FjoyMesg
Components List :

FJoyMesg : TFjoyMesg Form		Pboard : Tpanel of FjoyMesg	
Border Style	bsSingle	BorderStyle	bsSingle
Height	480	Caption	-None-
Name	FjoyMesg	Color	clWhite
Width	640	Height	393
-	-	Name	Pboard
-	-	Width	617
Ijet1: Timage of Pboard		Pcalibrate : Tpanel of Pboard	
AutoSize	True	BorderStyle	bsSingle
Height	135	Height	241
Name	Ijet1	Left	136
Picture	size 135 x 94 pixels	Name	Pcalibrate
Stretch	false	Visible	False
Widht	94	Width	415
Panel4 : Tpanel of Pcalibrate		Pcnt : Tpanel of Pcalibrate	
Caption	Notiy Range	Caption	3
Height	25	Height	25
Left	16	Left	112
Name	Panel4	Name	Pcnt
Top	16	Top	16
Width	97	Width	65
SpinButton1 : TSpinButton of Pcalibrate		BitBtn2 : TBitBtn of Pcalibrate	
Height	25	Caption	Cancel
Left	176	Height	25
Name	SpinButton1	Left	320
Top	16	Name	BitBtn2
Width	20	Top	16
-	-	Width	73
BitBtn3 : TBitBtn of Pcalibrate		BitBtn4 : TBitBtn of Pcalibrate	
Caption	Center a joystick then press this button	Caption	Move Joystick to upper left then press button
Height	33	Height	33
Left	16	Left	16
Name	BitBtn3	Name	BitBtn4
Top	64	Top	104
Width	377	Width	377

BitBtn5 : TBitBtn of Pcalibrate		BitBtn6 : TBitBtn of Pcalibrate	
Caption	Move Joystick to lower right the press button	Caption	Center Joystick then press button to confirm
Height	33	Height	33
Left	16	Left	16
Name	BitBtn5	Name	BitBtn6
Top	64	Top	184
Width	377	Width	377
Panel1 : Tpanel of FjoyMesg		SpeedButton1 : TSpeedButton of Panel1	
Caption	- None -	Caption	Start
Height	45	Height	25
Left	0	Left	8
Name	Panel1	Name	SpeedButton1
Top	406	Top	9
Width	632	Width	33
SpeedButton2 : TSpeedButton of Panel1		Px : Tpanel of Panel1	
Caption	Cablibrate	Caption	-None-
Height	25	Height	25
Left	48	Left	336
Name	SpeedButton2	Name	Px
Top	8	Top	8
Width	57	Width	73
Py : Tpanel of Panel1		Pbutton : Tpanel of Panel1	
Caption	- None -	Caption	- None -
Height	25	Height	25
Left	408	Left	480
Name	Py	Name	Pbutton
Top	8	Top	8
Width	73	Width	73
BitBtn1 : TBitBtn of Panel1			
Caption	Close		
Height	25		
Left	563		
Name	BitBtn1		
Top	9		
Width	57		

** - สำหรับ Properties อื่นที่ไม่กล่าวถึง คือไม่มีการเปลี่ยนแปลงใด ๆ

- 'of' หมายถึง สร้างอยู่ใน เช่น

Pboard : Tpanel of FjoyMesg หมายถึง Pboard มี Class เป็น Tpanel สร้างอยู่ใน FjoyMesg

- สำหรับ Events ของแต่ละ Objects แสดงอยู่ในลิสติงที่ 1

ลิสติงที่ 1

```
00001: unit uJoyMesg;
00002: interface
00003: uses
00004:   Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
00005:   MMSystem, MPlayer, Spin, ExtCtrls, StdCtrls, Buttons;
00006: const
00007:   MAXMOVERANGE = 100;
00008:   ACTIVE_TIMER = 10; (* msec. *)
00009: type
00010:   TFjoyMesg = class(TForm)
00011:     Panel1: TPanel;
00012:     SpeedButton1: TSpeedButton;
00013:     SpeedButton2: TSpeedButton;
00014:     BitBtn1: TBitBtn;
00015:     Px: TPanel;
00016:     Py: TPanel;
00017:     Pbutton: TPanel;
00018:     Pboard: TPanel;
00019:     Ijet1: TImage;
00020:     Pcalibrate: TPanel;
00021:     Bevel1: TBevel;
00022:     Bevel2: TBevel;
00023:     BitBtn2: TBitBtn;
00024:     BitBtn3: TBitBtn;
00025:     BitBtn4: TBitBtn;
00026:     BitBtn5: TBitBtn;
00027:     BitBtn6: TBitBtn;
00028:     SpinButton1: TSpinButton;
00029:     Pcnt: TPanel;
00030:     Panel4: TPanel;
00031:     { Joystick control part }
00032:     procedure JoyInit;
00033:     procedure JoyRelease;
00034:     procedure CalibrateCenter;
00035:     procedure CalibrateUpLeft;
00036:     procedure CalibrateDownRight;
00037:     function  JoyActivate( wndHdl: HWND; jid : integer ) : boolean;
00038:     procedure DoJoyPushButton( button : integer );
00039:     procedure DoJoyMovement( x,y : integer );
00040:     { Normal form }
00041:     procedure FormCreate(Sender: TObject);
00042:     procedure InitVar;
00043:     procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);
00044:     procedure SpeedButton1Click(Sender: TObject);
00045:     procedure SpeedButton2Click(Sender: TObject);
00046:     procedure SpinButton1UpClick(Sender: TObject);
00047:     procedure SpinButton1DownClick(Sender: TObject);
00048:     procedure BitBtn2Click(Sender: TObject);
00049:     procedure BitBtn3Click(Sender: TObject);
00050:     procedure BitBtn4Click(Sender: TObject);
00051:     procedure BitBtn5Click(Sender: TObject);
00052:     procedure BitBtn6Click(Sender: TObject);
00053:     procedure BitBtn1Click(Sender: TObject);
00054:   private
00055:     { Private declarations }
00056:     centerX, centerY : integer;
00057:     dNegX, dPosX : integer;
00058:     dNegY, dPosY : integer;
00059:     joyId : integer;
00060:     numOfJoyDev : integer;
00061:     joyCaps: TJoyCaps;
00062:     joyInfo : TJoyInfo;
00063:   public
00064:     { Public declarations }
00065:     notifyRange : integer;
00066:   protected
00067:     { Windows messages }
00068:     procedure MMJOY1MOVE(var Mesg: TMessage); message MM_JOY1MOVE;
00069:     procedure MMJOY1BUTTONDOWN(var Mesg: TMessage); message MM_JOY1BUTTONDOWN;
00070:     procedure MM_JOY1BUTTONUP(var Mesg: TMessage); message MM_JOY1BUTTONUP;
00071:   end;
00072: var
00073:   FjoyMesg: TFjoyMesg;
```

→

ลิสติงที่ 1 ต่อ

```
00074:  implementation
00075:  {$R *.DFM}
00076:  (*- Joystick Control Interface ----------*)
00077:  procedure TFjoyMesg.JoyInit;
00078:  begin
00079:    centerX:=32768; centerY:=32768; joyId:=0; NumOfJoyDev:=0; notifyRange:=3;
00080:    dNegX:=centerX div notifyRange; dPosX:=centerX div notifyRange;
00081:    dNegY:=centerX div notifyRange; dPosY:=centerX div notifyRange;
00082:  end;
00083:  procedure TFjoyMesg.JoyRelease;
00084:  begin
00085:    JoyReleaseCapture( joyId );
00086:  end;
00087:  procedure TFjoyMesg.CalibrateCenter;
00088:  var ret : integer; jInfo : TJoyInfo;
00089:  begin
00090:    ret:=joyGetPos( joyID, @jInfo);
00091:    if ( ret=JOYERR_NOERROR ) then
00092:    begin
00093:      centerX:=jInfo.wXpos;
00094:      centerY:=jInfo.wYpos;
00095:    end else
00096:    begin
00097:      centerX:=(joyCaps.wXmax-joyCaps.wXmin) div 2;
00098:      centerY:=(joyCaps.wYmax-joyCaps.wYmin) div 2;
00099:    end;
00100:  end;
00101:  procedure TFjoyMesg.CalibrateUpLeft;
00102:  var ret : integer; jInfo : TJoyInfo;
00103:  begin
00104:    ret:=joyGetPos( joyID, @jInfo);
00105:    if ( ret=JOYERR_NOERROR ) then
00106:    begin
00107:      dNegX:=(centerX-jInfo.wXPos) div notifyRange;
00108:      dNegY:=(centerY-jInfo.wYPos) div notifyRange;
00109:    end else
00110:    begin
00111:      dNegX:=centerX div notifyRange;
00112:      dNegY:=centerY div notifyRange;
00113:    end;
00114:  end;
00115:  procedure TFjoyMesg.CalibrateDownRight;
00116:  var ret : integer; jInfo : TJoyInfo;
00117:  begin
00118:    ret:=joyGetPos( joyID, @jInfo);
00119:    if ( ret=JOYERR_NOERROR ) then
00120:    begin
00121:      dPosX:=(jInfo.wXPos-CenterX) div notifyRange;
00122:      dPosY:=(jInfo.wYPos-CenterY) div notifyRange;
00123:    end else
00124:    begin
00125:      dPosX:=centerX div notifyRange;
00126:      dPosY:=centerY div notifyRange;
00127:    end;
00128:  end;
00129:  function TFjoyMesg.JoyActivate( wndHdl: HWND; jid : integer ) : boolean;
00130:  var ret : integer;
00131:  begin
00132:    JoyActivate:=false;
00133:    numOfJoyDev:=JoyGetNumDevs;
00134:    if ( numOfJoyDev>2 ) then numOfJoyDev:=2;
00135:    joyId:=jid;
00136:    ret:=JoyGetDevCaps( joyId, @joyCaps, sizeof( TJoyCaps ) );
00137:    if ( ret=JOYERR_NOERROR ) then
00138:    begin
00139:      ret:=JoyGetPos( joyId, @joyInfo );
00140:      if ( ret=JOYERR_NOERROR ) then
00141:      begin
00142:        ret:=joySetCapture( wndHdl, jid, ACTIVE_TIMER, FALSE );
00143:        if ( ret=JOYERR_NOERROR ) then
00144:        begin
00145:          JoyActivate:=true;
00146:        end;
```

→

ลิสติงที่ 1 ต่อ

```
00147:       end;
00148:     end;
00149:   end;
00150:   procedure TFjoyMesg.MMJOY1MOVE(var Mesg: TMessage);
00151:   var xPos,yPos, button : word; locX, locY: Integer;
00152:   begin
00153:     xPos:=loword( mesg.lparam );
00154:     yPos:=hiword( mesg.lparam );
00155:     button:=mesg.wparam;
00156:     if ( xPos<centerX ) then locX:=( XPos-centerX ) div dNegX
00157:     else locX:=( XPos-centerX ) div dPosX;
00158:     if ( YPos<CenterY ) then locY:=( centerY-YPos ) div dNegY
00159:     else locY:=( centerY-YPos ) div dPosY;
00160:     DoJoyMovement( locX, locY );
00161:     DoJoyPushButton( button );
00162:   end;
00163:   procedure TFjoyMesg.MMJOY1BUTTONDOWN(var Mesg: TMessage);
00164:   begin
00165:     (* Method to detect joystick button down,
00166:        lo byte of wparam is button id to press *)
00167:   end;
00168:   procedure TFjoyMesg.MM_JOY1BUTTONUP(var Mesg: TMessage);
00169:   begin
00170:     (* Method to detect joystick button up
00171:        lo byte of wparam is button id to release *)
00172:   end;
00173:   procedure TFjoyMesg.DoJoyPushButton( button : integer );
00174:   begin
00175:     Pbutton.caption:=IntToStr( button );
00176:   end;
00177:   procedure TFjoyMesg.DoJoyMovement( x,y : integer );
00178:   var jx,jy : integer;
00179:   begin
00180:     Px.caption:=IntToStr( x );
00181:     Py.caption:=IntToStr( y );
00182:     jx:=Ijet1.left; jy:=Ijet1.top;
00183:     jx:=jx+x; jy:=jy-y;
00184:     if ( jx<1 ) then jx:=1;
00185:     if ( jx>( Pboard.width-Ijet1.width ) ) then
00186:       jx:=Pboard.width-Ijet1.width;
00187:     if ( jy<1 ) then jy:=1;
00188:     if ( jy>( Pboard.height-Ijet1.height ) ) then
00189:       jy:=Pboard.height-Ijet1.height;
00190:     Ijet1.left:=jx; Ijet1.top:=jy;
00191:   end;
00192:   (*-------------------------------------*)
00193:   procedure TFjoyMesg.InitVar;
00194:   begin
00195:     JoyInit;
00196:   end;
00197:   procedure TFjoyMesg.FormCreate(Sender: TObject);
00198:   begin
00199:     with FjoyMesg do begin left:=0; top:=0; end;
00200:     with Ijet1 do begin left:=256; top:=120; end;
00201:     InitVar;
00202:   end;
00203:   procedure TFjoyMesg.FormCloseQuery(Sender: TObject; var CanClose: Boolean);
00204:   begin
00205:     joyRelease; canclose:=true;
00206:   end;
00207:   procedure TFjoyMesg.SpeedButton1Click(Sender: TObject);
00208:   begin
00209:     JoyActivate( FjoyMesg.handle, JOYSTICKID1 );
00210:     SpeedButton1.enabled:=false;
00211:   end;
00212:   procedure TFjoyMesg.SpeedButton2Click(Sender: TObject);
00213:   begin
00214:     with Pcalibrate do begin left:=104; top:=72; visible:=true; end;
00215:     Pcnt.caption:=IntToStr( notifyRange );
00216:   end;
00217:   procedure TFjoyMesg.SpinButton1UpClick(Sender: TObject);
00218:   var tmp : integer;
00219:   begin
```

→

ลิสติงที่ 1 ต่อ

```
00220:     tmp:=notifyRange;
00221:     inc( tmp ); if ( tmp>MAXMOVERANGE ) then tmp:=1;
00222:     Pcnt.caption:=IntToStr( tmp );
00223:     notifyRange:=tmp;
00224:   end;
00225:   procedure TFjoyMesg.SpinButton1DownClick(Sender: TObject);
00226:   var tmp : integer;
00227:   begin
00228:     tmp:=notifyRange;
00229:     dec( tmp ); if ( tmp<1 ) then tmp:=MAXMOVERANGE;
00230:     Pcnt.caption:=IntToStr( tmp );
00231:     notifyRange:=tmp;
00232:   end;
00233:   procedure TFjoyMesg.BitBtn2Click(Sender: TObject);
00234:   begin
00235:     BitBtn3.enabled:=true; BitBtn4.enabled:=false; BitBtn5.enabled:=false;
00236:     BitBtn6.enabled:=false; Pcalibrate.visible:=false;
00237:   end;
00238:   procedure TFjoyMesg.BitBtn3Click(Sender: TObject);
00239:   begin
00240:     CalibrateCenter; BitBtn3.enabled:=false; BitBtn4.enabled:=true;
00241:   end;
00242:   procedure TFjoyMesg.BitBtn4Click(Sender: TObject);
00243:   begin
00244:     CalibrateUpLeft; BitBtn4.enabled:=false; BitBtn5.enabled:=true;
00245:   end;
00246:   procedure TFjoyMesg.BitBtn5Click(Sender: TObject);
00247:   begin
00248:     CalibrateDownRight; BitBtn5.enabled:=false; BitBtn6.enabled:=true;
00249:   end;
00250:   procedure TFjoyMesg.BitBtn6Click(Sender: TObject);
00251:   begin
00252:     CalibrateCenter; BitBtn6.enabled:=false; BitBtn3.enabled:=true;
00253:     Pcalibrate.visible:=false;
00254:   end;
00255:   procedure TFjoyMesg.BitBtn1Click(Sender: TObject);
00256:   begin
00257:    FjoyMesg.close;
00258:   end;
00259:   end.
```

ในส่วน Private declarations จะกำหนดตัวแปรที่ใช้เป็น Global ชนิด private ประกอบไปด้วย

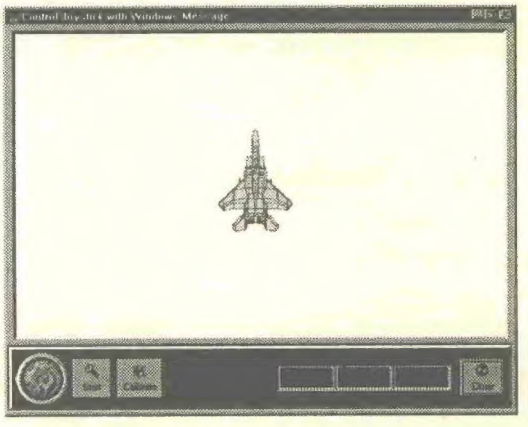
- joyId เก็บหมายเลข ID ของจอยสติ๊กที่ใช้อยู่ในปัจจุบัน
- numOfJoyDev เก็บจำนวนจอยสติ๊กที่สามารถต่อได้สูงสุด
- joyCaps เก็บข้อมูลส่วน device capabilities ของจอยสติ๊ก
- joyInfo เก็บสถานะปัจจุบันของจอยสติ๊กที่ใช้อยู่

สำหรับ centerX, centerY, dNegX, dPosX, dNegY, dPosY และ NotifyRange ได้กล่าวแล้วข้างต้น

Method ที่ใช้ควบคุมการทำงานของจอยสติ๊กจะอยู่ในส่วน Joystick Control Part ในช่วงบรรทัดที่ 32 ถึง 39 (ส่วน Declare) มีดังนี้ คือ

procedure TFjoyMesg.JoyInit เป็นส่วนการ Initialized ตัวแปรที่ใช้เกี่ยวกับการควบคุมจอยสติ๊กใน Class นี้

procedure TFjoyMesg.JoyRelease เป็นการยกเลิกการ Set Capture ของวินโดวส์ที่ส่งเมสเสจของจอยสติ๊กให้แก่ Form นี้



รูปที่ 3

procedure TFjoyMesg.CablirateCenter เป็นการหาจุดศูนย์กลางของจอยสติ๊กโดยอ่านตำแหน่งของจอยสติ๊กที่ตำแหน่งแป้นควบคุมอยู่ตรงกลางด้วยฟังก์ชัน JoyGetPos จากนั้นคำนวณหาตำแหน่งศูนย์กลางให้ CenterX เท่ากับค่าที่มากที่สุดของแกน X ลบค่าที่น้อยที่สุดในแกนเดียวกันแล้วหารสองเพื่อหาตำแหน่งตรงกลาง ส่วน CenterY ก็ใช้วิธีเดียวกันในการคำนวณ

procedure TFjoyMesg.CablirateUpLeft และ **procedure TFjoyMesg.CablirateDownRight** เป็นการหาตำแหน่งของมุมบนซ้าย และมุมล่างขวา ด้วยการแบ่งความละเอียดของช่วงความเคลื่อนไหวตามค่า NotifyRange เช่น สมมติว่าให้ NotifyRange เท่ากับ 4 ค่าที่ได้จากการเลื่อนปุ่มควบคุมทิศทางจะมีค่าอยู่ระหว่าง -3 ← 0 → 3 สำหรับทุกระนาบ

procedure TFjoyMesg.JoyActivate จะเรียกว่าเป็นส่วนการ Initialized ระบบจอยสติ๊กก็ได้ ฟังก์ชัน JoyGetNumDevs จะบอกจำนวนจอยสติ๊กที่สามารถใช้ในระบบได้ ฟังก์ชัน JoyGetDevCaps จะอ่านค่า Joystick Capabilities เพื่อใช้ในการคำนวณในส่วนการ Cablirate จอยสติ๊ก สำหรับฟังก์ชัน JoySetCapture เป็นการบอกให้วินโดวส์ส่งเมจเสจเกี่ยวกับจอยสติ๊กให้กับวินโดวส์ตามค่า Handle ของ WndHdl ตอบสนองตาม JoyID ที่กำหนดมาให้ และควบคุมความถี่ในการส่งเมสเสจจาก Active Timer

procedure TFjoyMesg.DoJoyPushButton(button : integer) เป็น Events ที่ตอบสนองการกดปุ่ม A, B, C, D ของจอยสติ๊กผ่านตัวแปร button

procedure TFjoyMesg.DoJoyMovement(x,y : integer); เป็น Events ตอบสนองการเคลื่อนที่เมื่อมีการกดปุ่มคีย์ลูกศร โดยค่า x และ y จะส่งค่าสัมพันธ์ในการเปลี่ยนแปลง เช่น เคลื่อนที่ไปทางขวา 3 หน่วยก็จะส่ง 3 มา และถ้าเคลื่อนที่ไปทางซ้าย 3 หน่วย ก็จะค่า -3 มาให้ เป็นต้น

สำหรับ Method ที่ใช้รับเมสเสจจากวินโดวส์เพื่อใช้ควบคุมจอยสติ๊กจะอยู่ช่วงบรรทัดที่ 67 ถึง 70 ดังนี้

procedure TFjoyMesg.MMJOY1MOVE(var Mesg: TMessage); message MM_JOY1MOVE; ใช้รับเมสเสจเกี่ยวกับจอยสติ๊กตัวที่ 1 ทั้งหมด โดยรับข้อมูลจากตัวแปร Mesg ซึ่งมี Field data คือ

Mesg.wparam : ส่งข้อมูลเกี่ยวกับปุ่มควบคุม ทั้ง 4 ปุ่ม

- ปุ่ม A ส่งค่าเท่ากับ 1
- ปุ่ม B ส่งค่าเท่ากับ 2
- ปุ่ม C ส่งค่าเท่ากับ 4
- ปุ่ม D ส่งค่าเท่ากับ 8

ถ้าเป็นการกดปุ่มร่วมเช่นกดปุ่ม A, B, C พร้อมกัน ก็จะส่งค่า 1 or 2 or 4 = 7 มาให้

Mesg.lparam : ส่งตำแหน่ง Physical ของตำแหน่งแกน X และ Y โดยบรรจุค่า

X อยู่ใน ไบต์คู่ต่ำ ของ Mesg.lparam สามารถอ่านค่าด้วย X:=LoWord(Mesg.lparam);

Y อยู่ใน ไบต์คู่สูง ของ Mesg.lparam สามารถอ่านค่าด้วย y:=HiWord(Mesg.lparam);

procedure TFjoyMesg.MMJOY1BUTTONDOWN(var Mesg: TMessage); message MM_JOY1BUTTONDOWN; ใช้รับเมสเสจเกี่ยวกับการกดปุ่ม A, B, C และ D ผ่านตัวแปร Mesg.wparam คือ

ไบต์ต่ำ ให้ค่าคีย์ที่กดหรือค่าคีย์ที่กดพร้อมกัน เช่น กดปุ่ม A ให้ค่า 1 หรือ กดปุ่ม A+B จะให้ค่า 3 สามารถอ่านค่าได้ด้วย Button:=Lo (Mesg.wparam);

ไบต์สูง ให้ค่าคีย์ที่กดล่าสุด เช่น ถ้ากดปุ่ม A, C, D และ B พร้อมกัน ก็จะได้ค่า 2 (คือ B)

procedure TFjoyMesg.MM_JOY1BUTTONUP(var Mesg: TMessage); message MM_JOY1BUTTONUP; ใช้รับเมสเสจเกี่ยวกับการกดปุ่ม A, B, C และ D ผ่านตัวแปร Mesg.wparam คือ

เมื่อกดปุ่มพร้อม ๆ กัน แล้วปล่อยปุ่มใดปุ่มหนึ่ง ค่าไบต์ต่ำของ Mesg.wparam จะให้ค่าที่ยังคงกดอยู่ เช่น เมื่อกดปุ่ม A, B, C และ D พร้อมกัน แล้วปล่อยปุ่ม C จะอ่านค่าได้คือ 11

ไบต์สูง จะให้ค่าที่ตรงกันข้ามกันคือ ให้ค่าคีย์ที่ปล่อย หลังจากการกดพร้อม ๆ กัน

จะเห็นได้ว่าการควบคุมจอยสติ๊กนั้นไม่ยากเลย เราสามารถรับรู้ถึงการเปลี่ยนแปลงของปุ่มต่าง ๆ ด้วย Method เพียง Method เดียวคือ **TFjoyMesg.MMJOY1MOVE(var Mesg: TMessage);** สำหรับ Method ที่รับเมสเสจอื่นที่กล่าวมาข้างต้นจะใช้เมื่อต้องการควบคุมแบบพิเศษ (Advance Control) ส่วนใหญ่ใช้ในการเขียนโปรแกรมเกมที่สามารถเล่นได้บนวินโดวส์ เป็นต้น

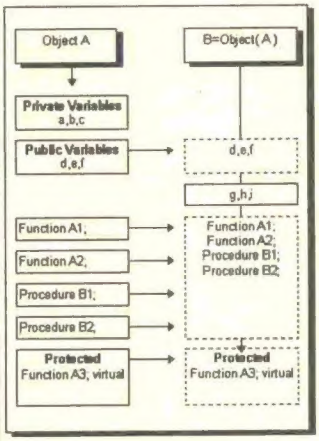
จากโปรแกรมข้างต้นนี้ เมื่อนำไปใช้งานจริงจะต้องมีการเขียนใหม่ทุกครั้งกับทุกโปรแกรมเมื่อต้องการติดต่อกับจอยสติ๊ก ทำให้ยุ่งยากและเสียเวลาในการพัฒนาโดยเปล่าประโยชน์ เราสามารถนำมาแปลงเป็น Objects ใช้หลักการของการโปรแกรมแบบ OOP (Object Oriented Programming) ซึ่งพอจะอธิบายให้เข้าใจง่าย ๆ ดังนี้

การโปรแกรมแบบ OOP เราต้องมองแต่ละส่วนของโปรแกรม (ประกอบไปด้วย ตัวแปร โปรแกรมย่อย) เป็นแบบ Objects เป็นชิ้นเป็นอัน (ถ้าจะให้เข้าใจง่าย คือคล้ายกับตัวต่อ LEGO ที่มีลักษณะเป็นชิ้น ๆ รูปทรงหลายแบบ สามารถนำมาต่อเป็นรูปร่างต่าง ๆ ได้) และการที่เราจะเข้าถึงตัวแปร (Variables) ต่าง ๆ ใน Objects นั้น ๆ ต้องผ่านกิจกรรมหรือโปรแกรมย่อย (เรียกอีกชื่อว่า Method) ที่สร้างขึ้นเพื่อเป็นกิจกรรมตอบสนองโดยเฉพาะ

โปรแกรมแบบ OOP มีหัวใจหลักอยู่ 3 ประการที่สำคัญคือ

- เอนแคปซูล (encapsulation) คือความเป็นเอกเทศ เป็นการรวมเอาตัวแปรและโปรแกรมย่อยเข้าไว้ด้วยกัน ทำให้สามารถปกป้องข้อมูลเหล่านั้นได้
- อินเฮอริต (inheritance) หมายถึงการสืบสกุล เป็นการถ่ายทอดความสามารถของ Objects แรก (ต้นสกุล) ซึ่งก็คือตัวแปรและกิจกรรมต่าง ๆ ไปยัง Objects ลูก

● โพลิมอร์ฟิซึม (polymorphism) หมายถึงการเลือกปฏิบัติทำให้สามารถเพิ่มขั้นความสามารถของกิจกรรมต่าง ๆ ที่เคยเกิดขึ้นแล้วเป็นกิจกรรมใหม่ที่มีความสามารถมากกว่าเดิม ตัวอย่างเช่น



Object A เป็น Object ต้นสกุล มีตัวแปรและโปรแกรมย่อย Declare ไว้ 3 ส่วน คือ Private เป็นส่วนที่ถือว่าเป็นการปกป้อง (encapsulation) Public เป็นส่วนที่อนุญาตให้ผู้สืบสกุลสามารถนำไปใช้ได้ (inheritance) และ Protect เป็นส่วนที่ให้เลือกปฏิบัติ (polymorphism)

Object B สืบสกุลจาก Object A ทำให้สามารถใช้ตัวแปรและโปรแกรมย่อยชนิด Public ของ Object A (inheritance) ได้ สำหรับ Function A3 เป็นกิจกรรมชนิด polymorphism ทำให้ผู้สืบทอดสามารถนำไปพัฒนาความสามารถใหม่โดยใช้ชื่อกิจกรรมเดิม ดังนั้น Object B จึงมีกิจกรรม A3 ใหม่ ทำให้การเรียกใช้กิจกรรม A3 ของ Object B เอง หรือ Object ที่สืบสกุลจาก Object B เรียกใช้กิจกรรม A3 นี้

ผู้ที่สนใจการโปรแกรมแบบ OOP จริง ๆ ขอแนะนำให้อ่านหนังสือ "การโปรแกรมแบบ OOP ด้วยภาษาปาสคาล ของ อาจารย์บุญเลิศ เอี่ยมทัศนา" ซึ่งมีตัวอย่างประกอบคำอธิบาย ทำให้เข้าใจได้เป็นอย่างดี

โปรแกรม

โปรแกรมควบคุมจอยสติ๊กแบบ OOP ได้แสดงไว้ในลิสติงที่ 2 โดยส่วนกิจกรรมที่อนุญาตให้ผู้สืบสกุลใช้งานแสดงไว้ในส่วน Public (จะไม่พูดถึงรายละเอียด เพราะคล้ายกับลิสติงที่ 1) สำหรับกิจกรรมที่สำคัญและต้องการให้ผู้สืบสกุลนำไปพัฒนาเพิ่มเติม ประกอบไปด้วย

procedure TJoyStick.DoJoyPushButton(button : integer); virtual; สังเกตว่ากิจกรรมนี้เป็นกิจกรรมว่างคือไม่ได้แสดงโค้ดใด ๆ จุดประสงค์เพื่อให้ Process ให้ค่าแก่พารามิเตอร์ button เตรียมไว้ให้กิจกรรมที่จะเกิดขึ้นในอนาคตจากการสืบสกุล จึง Declare ไว้เป็น Virtual ในส่วน Protected เพราะต้องการให้กิจกรรมนี้ตอบสนองใน Object นี้เท่านั้น

procedure TJoyStick.DoJoyMovement(x,y : integer); virtual; มีจุดประสงค์เช่นเดียวกันกับกิจกรรม DoJoyPushButton แต่เป็นการส่งพารามิเตอร์ตำแหน่ง X และ Y เมื่อมีการเลื่อนปุ่มควบคุมทิศทางของจอยสติ๊ก

สำหรับ Object ลูกแสดงไว้ในลิสติงที่ 3 รวมกับโปรแกรมที่ใช้รันจริง ในบรรทัดที่ 9 แสดงการสืบสกุลของ TJoyId1 สืบสกุลจาก TJoyStick มีกิจกรรมประกอบดังนี้

ลิสติงที่ 2

```
00001:  unit JoyObj;
00002:  interface
00003:  uses MMSystem, Windows, Messages, ExtCtrls, Classes;
00004:  type
00005:    TJoyStick = Object
00006:    public
00007:      constructor JoyInit( owner : TComponent );
00008:      destructor  JoyDone; virtual;
00009:      function    SetGamePort( jid : integer ) : boolean;
00010:      procedure   CalibrateCenter;
00011:      procedure   CalibrateUpLeft;
00012:      procedure   CalibrateDownRight;
00013:      function    Activate( jid : integer ) : boolean;
00014:      procedure   JoyProcess( Sender : TObject );
00015:      procedure   SetnotifyRange( nRange : integer );
00016:      function    GetnotifyRange : integer;
00017:    protected
00018:      procedure   DoJoyPushButton( button : integer ); virtual;
00019:      procedure   DoJoyMovement( x,y : integer ); virtual;
00020:    private
00021:        notifyRange : integer;
00022:        centerX, centerY : integer;
00023:        dNegX, dPosX : integer;
00024:        dNegY, dPosY : integer;
00025:        joyId : integer;
00026:        numOfJoyDev : integer;
00027:        joyCaps: TJoyCaps;
00028:        joyInfo : TJoyInfo;
00029:        jTimer : TTimer;
00030:    end;
```

→

ลิสติงที่ 2 ต่อ

```
00031:  implementation
00032:  Uses SysUtils;
00033:  constructor TJoyStick.JoyInit( owner : TComponent );
00034:  begin
00035:    centerX:=32768; centerY:=32768; joyId:=0; NumOfJoyDev:=0; notifyRange:=3,
00036:    dNegX:=8192; dPosX:=8192; dNegY:=8192; dPosY:=8192;
00037:    jTimer:=TTimer.Create( owner );
00038:    jTimer.enabled:=false;
00039:    jTimer.Interval:=50;
00040:    jTimer.OnTimer:=JoyProcess;
00041:    jTimer.enabled:=true;
00042:  end;
00043:  destructor TJoyStick.JoyDone;
00044:  begin
00045:    jTimer.Free;
00046:  end;
00047:  function TJoyStick.SetGamePort( jid : integer ) : boolean;
00048:  begin
00049:    SetGamePort:=false;
00050:    numOfJoyDev:=JoyGetNumDevs;
00051:    if ( numOfJoyDev>2 ) then numOfJoyDev:=2;
00052:    if ( jid>numOfJoyDev ) then exit;
00053:    joyId:=jid; SetGamePort:=true;
00054:  end;
00055:  procedure TJoyStick.CalibrateCenter;
00056:  var ret : integer; jInfo : TJoyInfo;
00057:  begin
00058:    ret:=joyGetPos( joyID, @jInfo);
00059:    if ( ret=JOYERR_NOERROR ) then
00060:    begin
00061:      centerX:=jInfo.wXpos;
00062:      centerY:=jInfo.wYpos;
00063:    end else
00064:    begin
00065:      centerX:=(joyCaps.wXmax-joyCaps.wXmin) div 2;
00066:      centerY:=(joyCaps.wYmax-joyCaps.wYmin) div 2;
00067:    end;
00068:  end;
00069:  procedure TJoyStick.CalibrateUpLeft;
00070:  var ret : integer; jInfo : TJoyInfo;
00071:  begin
00072:    ret:=joyGetPos( joyID, @jInfo);
00073:    if ( ret=JOYERR_NOERROR ) then
00074:    begin
00075:      dNegX:=(centerX-jInfo.wXPos) div notifyRange;
00076:      dNegY:=(centerY-jInfo.wYPos) div notifyRange;
00077:    end else
00078:    begin
00079:      dNegX:=centerX div notifyRange;
00080:      dNegY:=centerY div notifyRange;
00081:    end;
00082:  end;
00083:  procedure TJoyStick.CalibrateDownRight;
00084:  var ret : integer; jInfo : TJoyInfo;
00085:  begin
00086:    ret:=joyGetPos( joyID, @jInfo);
00087:    if ( ret=JOYERR_NOERROR ) then
00088:    begin
00089:      dPosX:=(jInfo.wXPos-CenterX) div notifyRange;
00090:      dPosY:=(jInfo.wYPos-CenterY) div notifyRange;
00091:    end else
00092:    begin
00093:      dPosX:=centerX div notifyRange;
00094:      dPosY:=centerY div notifyRange;
00095:    end;
00096:  end;
00097:  function TJoyStick.Activate( jid : integer ) : boolean;
00098:  var ret : integer;
00099:  begin
00100:    SetGamePort( jid );
00101:    Activate:=false;
00102:    ret:=JoyGetDevCaps( joyId, @joyCaps, sizeof( TJoyCaps ) );
00103:    if ( ret=JOYERR_NOERROR ) then
```

→

ลิสติงที่ 2 ต่อ

```
00104:    begin
00105:      ret:=JoyGetPos( joyId, @joyInfo );
00106:      if ( ret=JOYERR_NOERROR ) then
00107:      begin
00108:        Activate:=true;
00109:      end;
00110:    end;
00111:  end;
00112:  procedure TJoyStick.JoyProcess( Sender : TObject );
00113:  var ret : word; locX, locY: Integer;
00114:  begin
00115:    ret:=JoyGetPos( joyId, @joyInfo );
00116:    if ( ret=JOYERR_NOERROR ) then
00117:    begin
00118:      case joyInfo.wButtons of
00119:        JOY_BUTTON1 : DoJoyPushButton( JOY_BUTTON1 );
00120:        JOY_BUTTON2 : DoJoyPushButton( JOY_BUTTON2 );
00121:        JOY_BUTTON3 : DoJoyPushButton( JOY_BUTTON3 );
00122:        JOY_BUTTON4 : DoJoyPushButton( JOY_BUTTON4 );
00123:      end;
00124:      if ( joyInfo.wXPos<centerX ) then locX:=( joyInfo.wXPos-centerX ) div dNegX
00125:      else locX:=( joyInfo.wXPos-centerX ) div dPosX;
00126:      if ( joyInfo.wYPos<CenterY ) then locY:=( centerY-joyInfo.wYPos ) div dNegY
00127:      else locY:=( centerY-joyInfo.wYPos ) div dPosY;
00128:      DoJoyMovement( locX, locY );
00129:    end;
00130:  end;
00131:  procedure TJoyStick.SetnotifyRange( nRange : integer );
00132:  begin
00133:    notifyRange:=nRange;
00134:  end;
00135:  function TJoyStick.GetnotifyRange : integer;
00136:  begin
00137:    GetnotifyRange:=notifyRange;
00138:  end;
00139:  procedure TJoyStick.DoJoyPushButton( button : integer );
00140:  begin
00141:    (* reserved *)
00142:  end;
00143:  procedure TJoyStick.DoJoyMovement( x,y : integer );
00144:  begin
00145:    (* reserved *)
00146:  end;
00147:  end.
```

■

ลิสติงที่ 3

```
00001:  unit uTJoyObj;
00002:  interface
00003:  uses
00004:    Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
00005:    JoyObj, StdCtrls, Buttons, ExtCtrls, MMSystem, Spin;
00006:  const
00007:    MAXMOVERANGE = 100;
00008:  type
00009:    TJoyId1  = object( TJoyStick )
00010:      constructor JoyInit( owner : TComponent );
00011:      destructor  JoyDone; virtual;
00012:    protected
00013:      procedure   DoJoyPushButton( button : integer ); virtual;
00014:      procedure   DoJoyMovement( x,y : integer ); virtual;
00015:    end;
00016:    TFJoyObj = class(TForm)
00017:      Panel1: TPanel;
00018:      SpeedButton1: TSpeedButton;
00019:      BitBtn1: TBitBtn;
00020:      Px: TPanel;
00021:      Py: TPanel;
00022:      Pbutton: TPanel;
00023:      Pboard: TPanel;
```

→

ลิสติงที่ 3 ต่อ

```
00024:       SpeedButton2: TSpeedButton;
00025:       Ijet1: TImage;
00026:       Pcalibrate: TPanel;
00027:       BitBtn2: TBitBtn;
00028:       Bevel1: TBevel;
00029:       Bevel2: TBevel;
00030:       BitBtn3: TBitBtn;
00031:       BitBtn4: TBitBtn;
00032:       BitBtn5: TBitBtn;
00033:       BitBtn6: TBitBtn;
00034:       SpinButton1: TSpinButton;
00035:       Pcnt: TPanel;
00036:       Panel4: TPanel;
00037:       procedure FormCreate(Sender: TObject);
00038:       procedure InitVar;
00039:       procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);
00040:       procedure SpeedButton1Click(Sender: TObject);
00041:       procedure BitBtn1Click(Sender: TObject);
00042:       procedure SpeedButton2Click(Sender: TObject);
00043:       procedure BitBtn3Click(Sender: TObject);
00044:       procedure BitBtn4Click(Sender: TObject);
00045:       procedure BitBtn5Click(Sender: TObject);
00046:       procedure BitBtn6Click(Sender: TObject);
00047:       procedure BitBtn2Click(Sender: TObject);
00048:       procedure SpinButton1UpClick(Sender: TObject);
00049:       procedure SpinButton1DownClick(Sender: TObject);
00050:     private
00051:       { Private declarations }
00052:       joyStick : TjoyId1;
00053:       ntfRange : integer;
00054:     public
00055:       { Public declarations }
00056:     end;
00057:   var
00058:     FJoyObj: TFJoyObj;
00059:   implementation
00060:   {$R *.DFM}
00061:   (*-------------------------------------*)
00062:   constructor TJoyId1.JoyInit( owner : TComponent );
00063:   begin
00064:     inherited JoyInit( owner );
00065:   end;
00066:   destructor TJoyId1.JoyDone;
00067:   begin
00068:     inherited JoyDone;
00069:   end;
00070:   procedure TJoyId1.DoJoyPushButton( button : integer );
00071:   begin
00072:     FJoyObj.Pbutton.caption:=IntToStr( button );
00073:   end;
00074:   procedure TJoyId1.DoJoyMovement( x,y : integer );
00075:   var jx,jy : integer;
00076:   begin
00077:     FJoyObj.Px.caption:=IntToStr( x );
00078:     FJoyObj.Py.caption:=IntToStr( y );
00079:     with FJoyObj do
00080:     begin
00081:       jx:=Ijet1.left; jy:=Ijet1.top;
00082:       jx:=jx+x; jy:=jy-y;
00083:       if ( jx<1 ) then jx:=1;
00084:       if ( jx>( Pboard.width-Ijet1.width ) ) then
00085:         jx:=Pboard.width-Ijet1.width;
00086:       if ( jy<1 ) then jy:=1;
00087:       if ( jy>( Pboard.height-Ijet1.height ) ) then
00088:         jy:=Pboard.height-Ijet1.height;
00089:       Ijet1.left:=jx; Ijet1.top:=jy;
00090:     end;
00091:   end;
00092:   (*-------------------------------------*)
00093:   procedure TFJoyObj.FormCreate(Sender: TObject);
00094:   begin
00095:     with FJoyObj do begin left:=1; top:=1; end;
00096:     with Ijet1 do begin left:=256; top:=120; end;
```

→

ลิสติงที่ 8 ต่อ

```
00097:    InitVar;
00098:  end;
00099:  procedure TFJoyObj.InitVar;
00100:  begin
00101:    joyStick.JoyInit( FJoyObj );
00102:    ntfRange:=joyStick.GetNotifyRange;
00103:  end;
00104:  procedure TFJoyObj.FormCloseQuery(Sender: TObject; var CanClose: Boolean);
00105:  begin
00106:    joystick.JoyDone;
00107:    canclose:=true;
00108:  end;
00109:  procedure TFJoyObj.SpeedButton1Click(Sender: TObject);
00110:  begin
00111:    joyStick.Activate( JOYSTICKID1 );
00112:  end;
00113:  procedure TFJoyObj.BitBtn1Click(Sender: TObject);
00114:  begin
00115:    FJoyObj.close;
00116:  end;
00117:  procedure TFJoyObj.SpeedButton2Click(Sender: TObject);
00118:  begin
00119:    with Pcalibrate do begin left:=104; top:=72; visible:=true; end;
00120:  end;
00121:  procedure TFJoyObj.BitBtn3Click(Sender: TObject);
00122:  begin
00123:    joyStick.CalibrateCenter;
00124:    BitBtn3.enabled:=false;
00125:    BitBtn4.enabled:=true;
00126:  end;
00127:  procedure TFJoyObj.BitBtn4Click(Sender: TObject);
00128:  begin
00129:    joyStick.CalibrateUpLeft;
00130:    BitBtn4.enabled:=false;
00131:    BitBtn5.enabled:=true;
00132:  end;
00133:  procedure TFJoyObj.BitBtn5Click(Sender: TObject);
00134:  begin
00135:    joyStick.CalibrateDownRight;
00136:    BitBtn5.enabled:=false;
00137:    BitBtn6.enabled:=true;
00138:  end;
00139:  procedure TFJoyObj.BitBtn6Click(Sender: TObject);
00140:  begin
00141:    joyStick.CalibrateCenter;
00142:    BitBtn6.enabled:=false; BitBtn3.enabled:=true;
00143:    Pcalibrate.visible:=false;
00144:  end;
00145:  procedure TFJoyObj.BitBtn2Click(Sender: TObject);
00146:  begin
00147:    BitBtn3.enabled:=true;
00148:    BitBtn4.enabled:=false;
00149:    BitBtn5.enabled:=false;
00150:    BitBtn6.enabled:=false;
00151:    Pcalibrate.visible:=false;
00152:  end;
00153:  procedure TFJoyObj.SpinButton1UpClick(Sender: TObject);
00154:  var tmp : integer;
00155:  begin
00156:    tmp:=joyStick.GetNotifyRange;
00157:    inc( tmp ); if ( tmp>MAXMOVERANGE ) then tmp:=1;
00158:    Pcnt.caption:=IntToStr( tmp );
00159:    joyStick.SetNotifyRange( tmp );
00160:  end;
00161:  procedure TFJoyObj.SpinButton1DownClick(Sender: TObject);
00162:  var tmp : integer;
00163:  begin
00164:    tmp:=joyStick.GetNotifyRange;
00165:    dec( tmp ); if ( tmp<1 ) then tmp:=MAXMOVERANGE;
00166:    Pcnt.caption:=IntToStr( tmp );
00167:    joyStick.SetNotifyRange( tmp );
00168:  end;
00169:  end.
```

■

constructor TJoyId1.JoyInit(owner : TComponent); เป็นกิจกรรมที่ใช้ Initialized ค่าเบื้องต้น จะต้องเรียกกิจกรรม Init ของ TJoyStick Object ด้วยคำสั่ง inherited JoyInit(owner); เพื่อ Initialized ค่าเบื้องต้นของ Object แม่ โดยผ่าน handle ของ Tform ที่เรียกใช้ทางตัวแปร owner เพื่อสร้าง Timer ในการส่ง Notify Message เมื่อมีการเปลี่ยนแปลงค่าของจอยสติ๊ก

destructor TJoyId1.JoyDone; virtual; เป็นกิจกรรมที่ใช้ Release Object แม่

procedure TJoyId1.DoJoyPushButton(button : integer); virtual; เป็นกิจกรรมตอบสนองการกดปุ่ม button ด้วยการนำค่าพารามิเตอร์ button แสดงขึ้นหน้าจอ

procedure TJoyId1.DoJoyMovement(x,y : integer); virtual; เป็นกิจกรรมตอบสนองการกดปุ่มควบคุมด้วยการกำหนดตำแหน่งใหม่ให้กับ Image : Ijet1 ทำให้รูปภาพเคลื่อนที่ตามการเลื่อนปุ่มลูกศร และแสดงตำแหน่งขึ้นหน้าจอที่ component Px และ Py

ทดสอบโปรแกรม

TFJoyObj Form ใช้เป็นโปรแกรมทดสอบการควบคุมจอยสติ๊กด้วย TJoyID1 Object โดยการสร้างออบเจ็กต์ joystick : TjoyID1 และ Initialized ค่าในส่วนของกิจกรรม TFJoyObj.InitVar

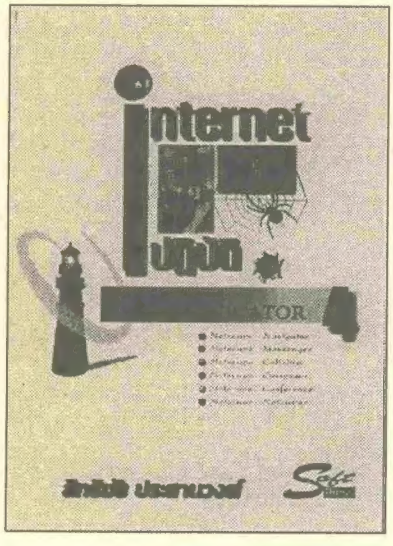
การควบคุมจอยสติ๊กจะเริ่มทำงานเมื่อมีการเรียกคำสั่ง joyStick. Activate(JOYSTICKID1); และส่งค่าจอยสติ๊ก JOYSTICKID1 ให้เพื่อระบุเป็นการใช้จอยส์ที่ 1 (ในระบบสามารถต่อจอยสติ๊กได้ 2 ตัว)

ออบเจ็กต์ joystick จะถูก release ทิ้งเมื่อทำการปิด Form ด้วยคำสั่ง joystick.JoyDone; ที่ใส่ไว้ในส่วน TFJoyObj.FormCloseQuery กิจกรรมอื่น ๆ หรือการสร้าง Form จะไม่กล่าวถึง เพราะมีส่วนคล้ายกับโปรแกรมในลิสติงที่ 1 ขอให้ศึกษาจากโปรแกรมข้างต้นเอง

สำหรับผู้ที่ไม่ต้องการเสียเวลาในการสร้าง Project นี้ สามารถใช้บริการซอฟต์เมล์ได้ โดยซอฟต์เมล์ของบทความนี้จัดทำเป็นโปรแกรม SETUP (ทันสมัยหน่อย) สามารถรันเพื่อทำการติดตั้งได้โดยตรง

สวัสดีครับ

INTERNET ปฏิบัติด้วย NETSCAPE COMMUNICATOR 4



หนังสือคู่มือการใช้โปรแกรม Netscape Communicator 4 ที่เขียนโดย คุณสิทธิชัย ประสานวงศ์ มีเนื้อหาเกี่ยวกับการใช้โปรแกรม Netscape Communicator, Netscape Messenger, Netscape Collabra, Netscape Composer, Netscape Conference และ Netscape Netcaster ภายในเล่มประกอบด้วยเนื้อหาเกี่ยวกับอินเทอร์เน็ตตั้งแต่ความรู้พื้นฐานทั่วไป การเตรียมเครื่องและการเลือกใช้บริการ การใช้โปรแกรมเพื่อท่องเวบ วิธีดาวน์โหลดข้อมูล การติดต่อสื่อสารผ่านอีเมล์ หรือประชุมทางไกล ตลอดจนเทคนิคต่าง ๆ ที่สามารถเรียนรู้และเรียกใช้งานได้ด้วยโปรแกรม Netscape

สนใจสามารถหาซื้อได้ตามร้านหนังสือชั้นนำทั่วไป หรือที่ร้านซีเอ็ดบุ๊คเซ็นเตอร์ทุกสาขา หรือจะสั่งซื้อโดยตรงได้ที่ บริษัท ซีเอ็ดยูเคชั่น จำกัด (มหาชน) 800/43-45 ซอยตระกูลสุข ถนนอโศก-ดินแดง ดินแดง กรุงเทพฯ 10320 โทร. 248-6280, 642-9800 แฟกซ์ 246-5999, 642-9866

ผู้แต่ง สิทธิชัย ประสานวงศ์
ISBN 974-7680-93-9
ราคา 320 บาท